Penerapan *Binary Search Algorithm* dalam Permainan Akinator

Mohammad Farhan Fahrezy - 13521106[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13521106@std.stei.itb.ac.id

***Abstract*—Akinator merupakan sebuah permainan berbasis web browser yang dirilis pada tahun 2007 oleh perusahaan asal Prancis, yaitu Elokence. Dalam permainan ini, pemain diberikan tugas untuk membayangkan suatu hal dengan cara memikirkan salah satu dari 3 opsi yang tersedia, yaitu karakter fiksi atau non-fiksi, objek atau binatang. Tugas dari Akinator adalah menebak hal yang pemain bayangkan dengan memberikan beberapa pertanyaan kepada player yang bisa dijawab dengan opsi *yes, no, don't know, probably*, dan *probably not*. Dalam aplikasinya, Akinator menggunakan konsep *Binary Search Tree* untuk menentukan hal apakah yang dibayangkan oleh pemain.**

Keywords*—Akinator, *Binary Search Tree*, *Binary Search Algorithm

I. PENDAHULUAN

Dewasa ini, ada banyak macam kegiatan yang bisa dilakukan untuk mengisi waktu luang. Salah satunya adalah memainkan permainan berbasis web. Permainan berbasis web adalah permainan yang bisa dimainkan dengan internet menggunakan web browser [1]. Permainan tersebut pada umumnya bersifat *free-to-play* atau bisa dimainkan secara gratis dan memiliki opsi untuk dimainkan secara *single player* maupun *multiplayer*. Beberapa permainan berbasis web juga memiliki adaptasi untuk dapat dimainkan di dalam aplikasi *mobile*, aplikasi desktop, maupun konsol. Bagi pemain, menggunakan permainan berbasis web memiliki beberapa keuntungan, seperti tidak harus menginstall suatu aplikasi secara spesifik untuk memainkan suatu permainan. Tetapi, pemainan berbasis web umumnya memiliki fitur yang kurang lengkap dan memiliki tampilan yang kurang sempurna jika dibandingkan dengan permainan yang diadaptasi ke dalam bentuk aplikasi.

Salah satu contoh dari permainan berbasis web adalah Akinator. Akinator merupakan sebuah permainan berbasis web yang dirilis pada tahun 2007 oleh perusahaan asal Prancis, yaitu Elokence. Permainan ini memiliki karakter utama seorang Jin yang bisa menebak isi pikiran pemain melalui pertanyaan yang diberikan oleh Akinator. Permainan ini termasuk ke dalam permainan bergenre *twenty questions*. *Twenty questions* merupakan salah satu genre permainan yang mendorong penalaran dan kreativitas induktif. Genre ini berasal dari Amerika Serikat dan mulai menyebar di abad ke-19[2] dan popularitasnya semakin meningkat selama tahun 1940-an ditandai dengan banyaknya radio pada saat itu menyiarkan permainan tersebut.

Permainan Akinator kembali *booming* saat salah satu pemilik kanal Youtube berkebangsaan Swedia, Felix Arvid Ulf Kjellberg, menggunggah *gameplay* Akinator ke dalam kanalnya, yaitu PewDiePie pada tahun 2015. Semenjak saat itu, popularitas Akinator semakin melesat beriringan dengan *content creator* lain yang ingin mencoba memainkan permainan tersebut.

Gambar 1.1 Tampilan Homescreen dari Website en.akinator.com
Sumber : https://en.akinator.com/

II. TEORI DASAR

A. Graf

1. Definisi Graf

Graf adalah struktur diskrit yang terdiri dari *vertices* (simpul) dan *edges* (sisi) yang menghubungkan *vertices* tersebut. Graf umumnya digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antara objek tersebut. Secara definisi, sebuah graf dapat didefinisikan sebagai berikut,

$$G = (V,E)$$

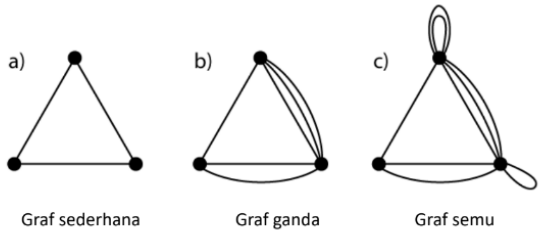
V merupakan himpunan tidak kosong dari simpul-simpul (*vertices*). V = {*v1,v2,…,vn*}.

E merupakan himpunan sisi (*edges*) yang menghubungkan sepasang simpul. E = {*e1,e2,…,en*}.

2. Jenis Graf

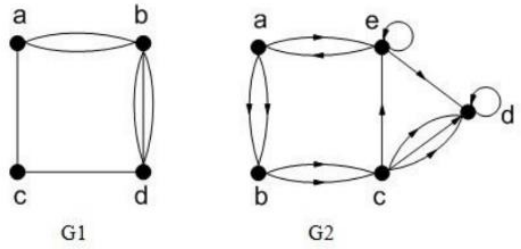
Berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, maka graf dapat digolongkan menjadi dua jenis, yaitu graf sederhana (*simple graph*) dan graf tak-sederhana (*unsimple*

graph). Graf sederhana adalah graf yang tidak mengandung gelang maupun sisi ganda. Graf tak-sederhana terbagi lagi menjadi dua jenis, yaitu graf ganda (*multi-graph*) dan graf semu (*pseudo-graph*). Graf ganda adalah graf yang mengandung sisi ganda. Graf semu adalah graf yang mengandung sisi gelang.



Gambar 2.1 a) Graf Sederhana, b) Graf Ganda, c) Graf Semu.
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf

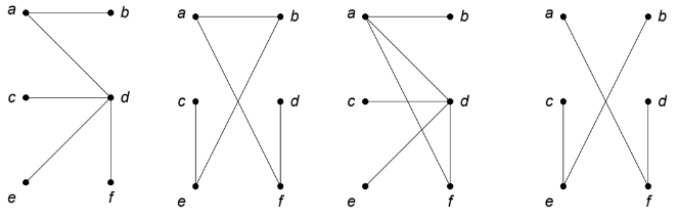
Berdasarkan orientasi arah pada sisi, graf dibagi menjadi 2 jenis, yaitu graf tak-berarah (*undirected graph*) dan graf berarah (*directed graph*).



Gambar 2.2 a) Graf tak-berarah, b) Graf Berarah.
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf

B. Pohon

Pohon adalah graf tak-berarah terhubung yang tidak mengandung sirkuit atau loop. Sebuah pohon juga dapat didefinisikan sebagai *G = (V,E)*. Sebuah graf tak-berarah merupakan sebuah pohon jika dan hanya jika terdapat satu jalur unik dari satu simpul ke simpul lainnya. Dalam sebuah pohon, terdapat sebuah simpul yang disebut sebagai akar (*root*). Setiap akar akan memiliki daun (*leaf*) yang dapat membentuk suatu pohon baru atau biasa disebut sebagai sub-pohon.

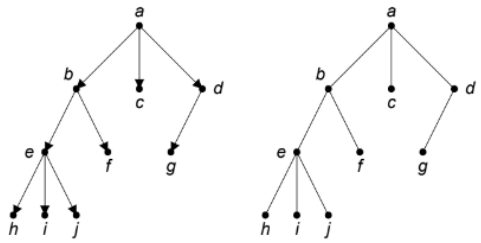


Gambar 2.3 a) Pohon, b) Pohon, c) Bukan Pohon, d) Bukan Pohon.
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf

C. Pohon Berakar

1. Definisi Pohon Berakar

Pohon berakar atau *rooted tree* merupakan sebuah pohon yang simpulnya diperlakukan sebagai akar dan sisinya diberi arah menuju daun sehingga menjadikan sebuah graf berarah.

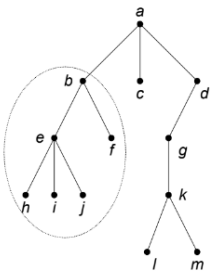


Gambar 2.4 a) Pohon Berakar, b) Pohon Berakar tanpa Arah.
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf

2. Terminoloagi Pohon Berakar

Pohon berakar memiliki beberapa terminologi sehingga dapat dibedakan dengan bentuk pohon lainnya. Terminologi pohon berakar adalah sebagai berikut,

a) Anak (*child*) dan Orangtua (*parent*)
Jika dua buah simpul dihubungkan dengan sebuah sisi, maka simpul asal disebut sebagai orangtua dan simpul asal disebut sebagai anak.
b) Saudara (*sibling*)
Jika n buah simpul memiliki orangtua yang sama, maka n buah simpul tersebut merupakan saudara.
c) Lintasan (*path*)
Lintasan adalah jalur yang dilalui diantara dua simpul.
d) Upapohon (*subtree)*
Upapohon merupakan sebuah pohon baru yang akarnya merupakan anak dari sebuah simpul.

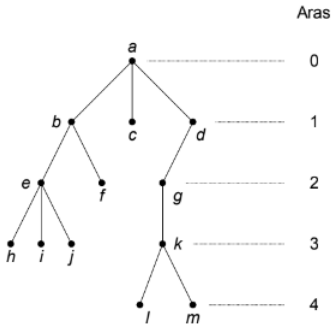


Gambar 2.5 Upapohon.
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag2.pdf

e) Derajat (*degree*)
Derajat sebuah simpul adalah jumlah upapohon atau anak yang dimiliki suatu simpul.
f) Daun (*leaf*)
Daun merupakan sebuah simpul yang berderajat nol atau tidak memiliki anak.
g) Simpul Dalam (*internal nodes*)
Simpul dalam merupakan suatu simpul yang memiliki

anak.

h) Aras (*level*) atau Tingkat
Aras adalah jumlah lintasan suatu simpul dihitung dari akar sebuah pohon.

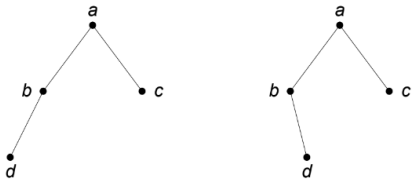


Gambar 2.6 Aras Sebuah Pohon.
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag2.pdf

i) Kedalaman (*depth*) atau Tinggi (*height*)
Kedalaman merupakan nilai maksimum dari aras sebuah pohon.

D. Pohon Biner

Pohon biner atau *binaty tree* merupakan salah satu contoh bentuk pohon berakar. Jenis pohon ini memiliki banyak aplikasi dalam dunia *computer science*. Setiap simpul pada pohon biner memiliki maksimum dua derajat. Anak yang dimiliki sebuat simpul dibedakan antara anak kiri (*left child*) dan anak kanan (*right child*) yang membuat pohon biner masuk ke dalam pohon terurut.



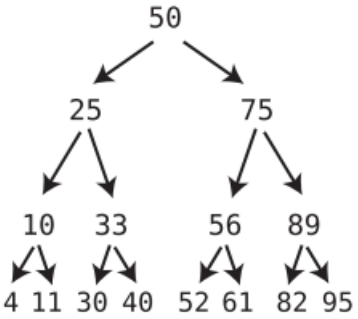
Gambar 2.7 Dua Buah Pohon Biner yang Berbeda.
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag2.pdf

Berdasarkan ada atau tidaknya anak dari sebuah simpul, setiap simpul dapat dibagi menjadi empat, yaitu simpul *uner left*, simpul *uner right*, simpul biner dan daun. Simpul *uner left* merupakan simpul yang hanya memiliki anak kiri. Simpul *uner right* merupakan simpul yang hanya memiliki anak kanan. Simpul biner merupakan simpul yang memiliki anak kiri dan anak kanan. Sedangkan simpul daun adalah simpul yang tidak memiliki anak.

E. Binary Search Tree

Dalam bahan Algoritma dan Struktur Data, ada banyak struktur data yang bisa digunakan dari bentuk pohon. Salah satu aplikasi dari struktur data pohon biner adalah *binary search tree*. *Binary search tree* memiliki struktur data yang sama dengan pohon biner, yaitu memiliki maksimum satu anak kiri dan satu anak kanan.

Dalam *binary search tree*, suatu simpul yang memiliki anak kiri, nilai yang disimpan dalam anak kiri dan keturunannya akan selalu lebih kecil daripada simpul itu sendiri. Sedangkan untuk anak kanan, nilai yang disimpan dalam anak kanan beserta keturunannya akan selalu lebih besar daripada simpul itu sendiri.



Gambar 2.8 Contoh Struktur Data *Binary Search Tree*.
Sumber: Wengrow, J., A Common-Sense Guide to Data Structures and Algorithms, Second Edition: Level Up Your Core Programming Skills (2nd ed.), Pragmatic Bookshelf, 2020, pp. 251

Algoritma *binary search tree* umumnya digunakan untuk mencari suatu nilai dari N buah data. Algoritma ini sangat efektif jika digunakan dalam kasus *average-case scenario* karena hanya membutuhkan log(N) langkah.

Cara kerja algoritma *binary search tree* untuk mencari suatu nilai x adalah dengan memilih suatu *current node* (atau biasanya akar) dan membandingkan nilai dalam simpul tersebut. Jika nilai dalam simpul tersebut sama dengan nilai x, maka algoritma selesai. Jika nilai dalam simpul tersebut tidak sama, maka ada tiga kondisi yang mungkin terjadi.

Kondisi pertama adalah nilai x lebih besar daripada nilai simpul. Jika kondisi ini terjadi, *current node* akan berpindah ke anak kanan dari *current node* awal dan dilakukan pengecekan ulang. Kondisi selanjutnya adalah ketika nilai x lebih kecil daripada nilai simpul. Jika kondisi ini terjadi, *current node* akan berpindah ke anak kiri dari *current node* awal dan dilakukan pengecekan ulang. Kondisi terakhir adalah kondisi tidak ditemukan. Kondisi ini dapat tercapai jika *current node* sudah tidak memiliki anak lagi atau jika *current node* memiliki nilai yang lebih kecil dari nilai x tetapi *current node* merupakan simpul *uner left* atau jika *current node* memiliki nilai yang lebih besar dari nilai x tetapi *current node* merupakan simpul *uner right.*

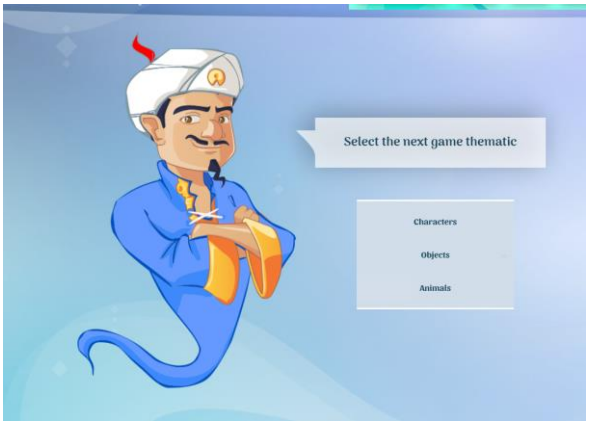
F. Akinator

Akinator merupakan permainan berbasis web yang pertama kali diluncurkan pada tahun 2007 oleh perusahaan asal Prancis, yaitu Elokence. Genre yang dimiliki oleh permainan ini adalah genre *twenty questions* yang dapat dimainkan tanpa memiliki *prior knowledge* tentang genre tersebut. Permainan ini sangat *booming* di tahun 2015 saat salah satu pemilik kanal Youtube berkebangsaan Swedia, Felix Arvid Ulf Kjellberg, menggunggah *gameplay* Akinator ke dalam kanalnya, yaitu

PewDiePie.

Akinator memiliki daya tariknya tersendiri kepada pemain. Daya tarik tersebut muncul dari cara kerja permainan tersebut yang seakan-akan dapat menebak isi pikiran pemain. Permainan ini dapat dimainkan secara gratis dalam website resminya, yaitu di https://en.akinator.com/.

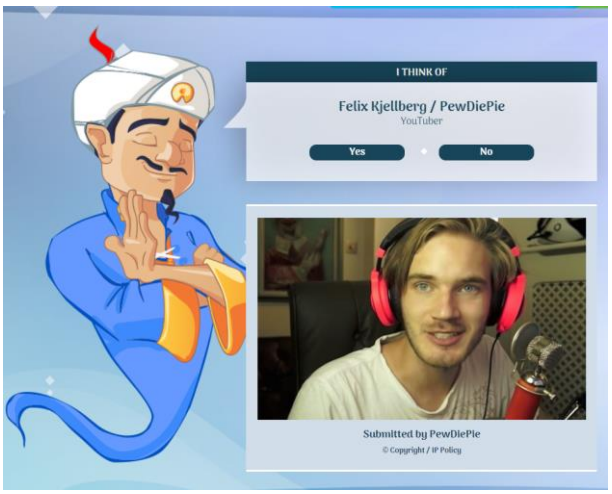
Alur bermain dari permainan ini adalah, pertama permain diminta untuk memilih diantara satu dari tiga kategori yang tersedia, yaitu *character*, *object*, atau *animal*.



Gambar 2.9 Akinator Meminta Kategori yang Akan Ditebak
Sumber : https://en.akinator.com/

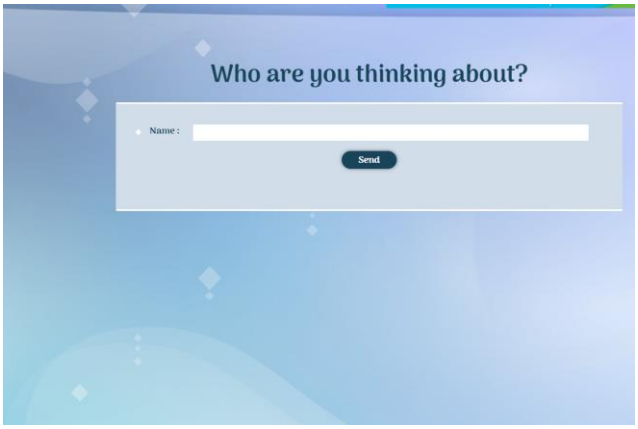
Setelah pemain memilih salah satu diantara tiga kategori tersebut, Akinator akan memberikan pertanyaan yang umum mengenai topik yang pemain bayangkan. Pertanyaan tersebut dapat dijawab dengan lima opsi, yaitu *yes*, *no*, *don't know*, *probably* dan *probably not*. Tiap jawaban yang pemain berikan akan menentukan banyaknya pertanyaan yang akan Akinator berikan.

Umumnya, Akinator akan mendapatkan jawaban dari topik yang pemain bayangkan dalam dua puluh (20) pertanyaan. Tetapi, jika topik yang kita ajukan tidak cukup umum ditanyakan, pertanyaan yang diberikan Akinator akan semakin banyak tanpa ada batasan tertentu.



Gambar 2.10 Hasil Tebakan Akinator Setelah 15 Pertanyaan
Sumber : https://en.akinator.com/

Jika tebakan yang diberikan Akinator belum tepat dengan apa yang pemain bayangkan, pemain dapat menekan tombol *no* dan Akinator akan melanjutkan memberikan pertanyaan yang menjurus kepada topik yang kita bayangkan. Pemain dapat terus diberikan pertanyaan sampai batas yang tidak berhingga sampai Akinator menemukan tebakan yang tepat atau pemain memilih berhenti untuk bermain. Jika pemain memutuskan untuk berhenti bermain, Akinator akan meminta pemain untuk memberikan jawaban dari topik yang kita bayangkan.



Gambar 2.11 Akinator Meminta Jawaban dari Pemain
Sumber : https://en.akinator.com/

III. Aplikasi Binary Search Algorithm dalam Permainan Akinator

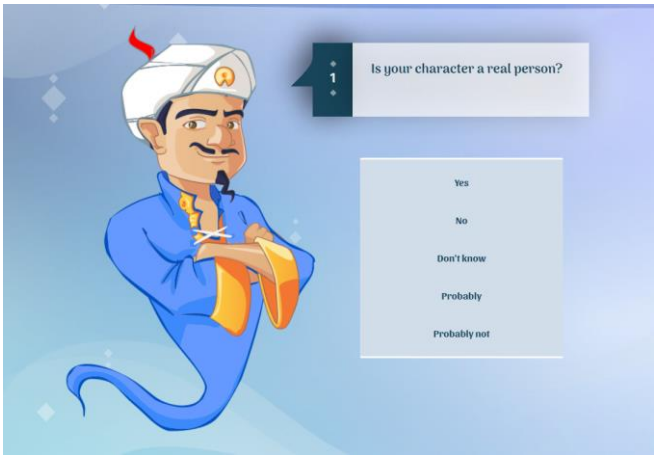
Binary search algorithm berkerja dengan membandingkan suatu nilai masukan dengan nilai yang dimiliki suatu simpul di dalam suatu pohon. Dalam setiap eksekusinya, algoritma ini membuang sebanyak N/2 kemungkinan, sehingga dalam keadaan *average case scenario*, yaitu saat data yang dicari berada tersebar di dalam jangkauan *bell* curve, algoritma ini sangat efisien untuk digunakan.

Felix Arvid Ulf Kjellberg	A Character Have two legs A Youtuber A real life person Male Alive ....
Christopher Columbus	A Character Have two legs discoverer of the american continent Male A real life person deceased ....
Monkey D Luffy	A Character Have two legs A Pirate Can strech his body ...
Naruto Uzumaki	A Character Have two legs A Ninja Can clone himself ...
Mohammad Farhan Fahre	A Character An undergraduate student A real life person Currently study in ITB ...

Gambar 3.1 Contoh Database Akinator [6]
Sumber : Dokumentasi Pribadi

Dalam permainan Akinator, algoritma yang digunakan untuk mencari tebakan memiliki konsep yang sama dengan *binary search algorithm*. Tetapi, ada sedikit perbedaan dengan algoritma tersebut, dimana *binary search algorithm* secara spesifik menggunakan tipe data pohon dan hanya bisa membandingan jawaban tergantung dari nilai integer yang dimiliki. Sedangkan dalam permainan Akinator, algoritma memanfaatkan suatu database yang dimiliki oleh Akinator dan memberikan tebakan akhir berdasarkan jawaban yang diberikan oleh pemain.

Cara kerja algoritma *binary search algorithm Akinator* (kedepannya saya sebut algoritma Akinator) adalah dengan memilih suatu fakta yang dapat menyisihkan setengah dari total data yang dimiliki oleh database. Sebagai contoh, pada gambar 3.1, terlihat bahwa ada lima data datum yang dimiliki oleh database. Dari kelima data tersebut, akan dipilih salah satu fakta yang hanya dimiliki oleh setengah dari polulasi dataset. Pernyataan yang bisa dijadikan pertanyaan oleh Akinator berdasarkan gambar 3.1 adalah "*is your character a real life person?"*



Gambar 3.2 Pertanyaan Akinator Berdasarkan Pemilihan Pertanyaan dari Database
Sumber : https://en.akinator.com/

Jika pemain memilih opsi *no*, maka algoritma Akinator akan langsung membuang semua data yang memiliki fakta "*A real life person"*. Sedangkan jika pemain memilih opsi *yes*, maka algoritma Akinator akan melakukan hal yang sebaliknya, yaitu membuang semua data yang tidak memiliki fakta fakta "*A real life person"*.

Gambar 3.3 a) Hasil Database saat Pemain Memilih *no* (kiri), dan b) Hasil Database saat Pemain Memilih *yes* (kanan)
Sumber : Dokumentasi Pribadi

Jika pemain memilih opsi *probably*, algoritma Akinator akan langsung melanjutkan pencarian fakta yang lebih condong ke data yang dituju tanpa melakukan penghapusan data. Contohnya, jika pemain memilih jawaban *probably*, maka Akinator akan memberikan pertanyaan lanjutan hanya berdasarkan fakta yang dimiliki oleh data pada gambar 3.3.b, yaitu kumpulan data yang memiliki fakta "*A real life person"*.

Sebaliknya, jika pemain memilih opsi *probably not*, algoritma Akinator akan langsung melanjutkan pencarian fakta yang lebih condong ke data yang dituju tanpa melakukan penghapusan data. Contohnya, jika pemain memilih jawaban *probably not*, maka Akinator akan memberikan pertanyaan lanjutan hanya berdasarkan fakta yang dimiliki oleh data pada gambar 3.3.a, yaitu kumpulan data yang tidak memiliki fakta "*A real life person"*.

Selanjutnya, jika pemain memilih opsi *don't know*, algoritma Akinator akan langsung melanjutkan pencarian fakta secara normal, tanpa melakukan operasi apapun. Opsi *don't know* jika dilihat dalam segi algoritma, merupakan opsi untuk melakukan *continue*. Karena dari opsi tersebut, tidak ada kesimpulan yang bisa diambil dan akan melanjutkan pencarian hingga didapatkan satu jawaban akhir.

Jika pada suatu kondisi, jumlah data yang dimiliki dataset semakin sedikit, algoritma Akinator bisa memilih pertanyaan yang lebih spesifik mengarah ke salah satu data. Contohnya, pada gambar 3.3.b, tersisa tiga data yang dapat dipilih. Akinator akan mencoba memilih fakta yang lebih menjurus ke salah satu kandidat, yaitu Felix Arvid Ulf Kjellberg dengan faktanya adalah "*A Youtuber"*.

Gambar 3.4 Contoh Pertanyaan Akinator Berdasarkan Fakta "*A Youtuber"*
Sumber : https://en.akinator.com/

Jika pemain memilih jawaban *yes*, maka algoritma Akinator akan langsung memberikan jawaban yang sesuai dengan database, yaitu Felix Arvid Ulf Kjellberg. Tetapi, jika seandainya tebakan dari algoritma Akinator bukanlah jawaban akhir yang diharapkan pemain, maka algoritma Akinator akan melakukan membuang jawaban yang diberikan dari database dan melanjutkan permainan dengan memberikan pertanyaan lanjutan.

Jika pada suatu kondisi algoritma Akinator tidak dapat memberikan jawaban yang tepat, maka algoritma Akinator akan meminta pemain untuk memberikan input mengenai topik apa yang sedang dibahas. Setelah itu, algoritma Akinator akan menambahkan datum yang diterima oleh pemain ke database Akinator agar pemain selanjutnya dapat memainkan Akinator dan mendapatkan tebakan yang lebih baik.

Felix Arvid Ulf Kjellberg	A Character Have two legs A Youtuber A real life person Male Alive ....
Christopher Columbus	A Character Have two legs discoverer of the american continent Male A real life person deceased ....
Monkey D Luffy	A Character Have two legs A Pirate Can strech his body ...
Naruto Uzumaki	A Character Have two legs A Ninja Can clone himself ...
Mohammad Farhan Fahr	A Character An undergraduate student A real life person Currently study in ITB ...
Mr Beast	A Character Have two legs A Youtuber A real life person Male Alive ....

Gambar 3.5 Algoritma Akinator Menambahkan Data Baru
Sumber : Dokumentasi Pribadi [6]

IV. KESIMPULAN

Binary Search Algorithm memiliki banyak aplikasi yang bisa digunakan untuk kehidupan sehari-hari. Salah satu contohnya adalah permainan berbasis web, Akinator. Permainan ini merupakan permainan bergenre *twenty questions* yang menerima input dari pemain untuk menentukan tebakan dari pikiran pemain. Secara garis besar, alur program Akinator dapat dijelaskan menggunakan salah satu konsep dari matematika diskrit, yaitu Pohon Biner. Alur program Akinator berkerja dengan prinsip yang mirip dengan *binary search algorithm* sehingga pemodelan algoritmanya bisa menggunakan *binary search algorithm*. Walaupun dalam praktek sebenarnya algoritma yang digunakan sedikit berbeda, algoritma tersebut dapat menjelaskan bagaimana keputusan untuk memilih jawaban berdasarkan input dari pemain dapat diambil.

V. UCAPAN TERIMAKASIH

Puji Syukur kepada Tuhan Yang Maha Esa karena berkat rahmat dan karunianya penulis dapat menyelesaikan makalah Mata Kuliah IF2120 sebagai bentuk tugas akhir dengan baik dan tepat pada waktunya. Penulis menyampaikan terima kasih kepada Dr. Nur Ulfa Maulidevi, S.T, M.Sc., selaku dosen pengajar Mata Kuliah IF2120 Kelas 01, beserta dosen penyampu mata kuliah lainnya yang juga telah membimbing kami dalam proses belajar mengajar. Penulis juga berterima kasih kepada orang tua yang telah berdoa, mendukung, dan memberikan motivasi untuk selalu belajar, sehingga penulis dapat menyelesaikan makalah ini. Tak lupa juga penulis berterima kasih kepada pembuat referensi yang penulis gunakan sehingga membantu penulis untuk menyelesaikan makalah IF2120 ini. Terakhir, penulis berterima kasih kepada semua pihak teman dan kolega yang menjadi partner diskusi dalam membantu penulis menyusun makalah ini dari awal hingga akhir.

REFERENCES

[1] D Schultheiss: Long-term motivations to play MMOGs: A longitudinal study on motivations, experience and behavior, page 344. DiGRA, 2007. [Accessed 10/12/2022 22.05]

[2] Walsorth, Mansfield Tracy. Twenty Questions: A Short Treatise on the Game, Holt, 1882. [Accessed 10/12/2022 22.05]

[3] Wengrow, J., A Common-Sense Guide to Data Structures and Algorithms, Second Edition: Level Up Your Core Programming Skills (2nd ed.), Pragmatic Bookshelf, 2020. [Accessed 12/12/2022 01.14]

[4] Munir, Rinaldi, 2021. Graf (Bagian 1). https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf [Accessed 12/12/2022 01.14]

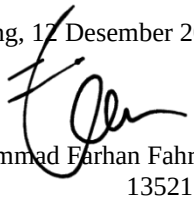
[5] Munir, Rinaldi, 2021. Pohon (Bagian 2). https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag2.pdf [Accessed 12/12/2022 01.14]

[6] Koder, Addy. How to make an app like the Akinator game in Python from scratch. 2021. https://youtu.be/lOfyN7zFI5s [Accessed 12/12/2022 01.14]

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 12 Desember 2022



Mohammad Farhan Fahrezy
13521106